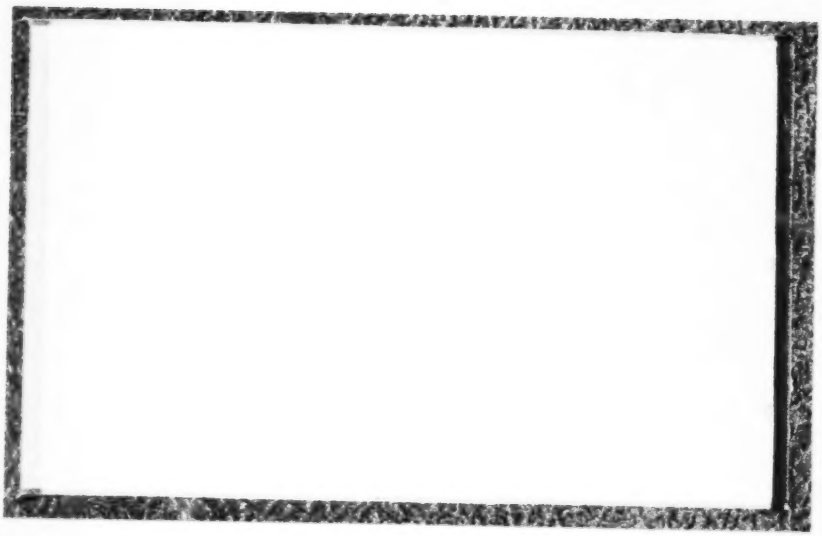
UNIVERSITÉ DE MONTRÉAL



CENTRE DE RECHERCHE SUR LES TRANSPORTS

CASE POSTALE 6128, SUCCURSALE A, MONTRÉAL, QUÉBEC, H3C 3J7

ROUTING WITH TIME WINDOWS
BY COLUMN GENERATION

by

Jacques Desrosiers, François Soumis
and Martin Desrochers

Ce rapport a été publié grâce à une subvention du fonds F.C.A.C. pour l'aide et le soutien à la recherche.

Université de Montréal

Centre de recherche sur les transports - Publication #277A
Ecole des Hautes Etudes Commerciales -
Cahiers du GERAD G-83-15

Août 1983, révisé en janvier 1984

Dépôt légal - 1er trimestre 1984

ABSTRACT

Consider a set of trips where each trip is specified a priori by a place of origin, a destination, a duration, a cost and a time interval within which the trip must begin. The trips may include visits to one or more specific points. Our problem is to determine the number of vehicles required together with their routes and schedules, so that each trip begins within its given time interval, while the fixed costs related to the number of vehicles, and the travel costs between trips are minimized. The problem is a generalization of the m-travelling salesman problem.

We use column generation on a set partitioning problem solved by simplex and branch-and-bound ; columns are generated by a shortest path algorithm with time windows on the nodes. Numerical results for several school bus transportation problems with up to 151 trips are discussed.

- - - - - - - - - - - - - - - -

RÉSUMÉ

Le problème considère un ensemble de parcours ayant chacun un lieu de début, un lieu de fin, une durée, un coût et un intervalle de temps pendant lequel ils doivent être effectués. Les parcours sont donnés a priori et peuvent consister en la visite d'un ou de plusieurs points. Il faut trouver le nombre, l'horaire et les routes de véhicules pour effectuer chacun des parcours dans l'intervalle de temps imposé et ce en minimisant les coûts de déplacement entre les parcours. Ce problème est une généralisation du m voyageur de commerce.

Nous avons mis au point une méthode optimale traitant le temps comme une variable continue et trouvant simultanément les horaires et les routes de véhicules. La méthode procède par génération de colonnes produites par un sous-problème de chemin le plus court avec contraintes de temps. Des résultats numériques sont présentés.

1. INTRODUCTION

This paper presents a method for optimally solving a fleet routing and scheduling problem. Before describing the problem, some terminology is introduced. A trip is a productive journey which must be carried out by a vehicle. The trip i is characterized by a place of origin, a destination, a duration, a cost and a time interval $[a_i,b_i]$ during which the trip must begin. An intertrip arc is an unproductive (i.e. empty) journey carried out by a vehicle. The intertrip arc (i,j) goes from the end of trip i to the beginning of trip j. Its duration t_{ij} and its cost c_{ij} may include respectively the duration and cost of the trip i. A route is a sequence of trips and intertrip arcs carried out by the same vehicle.

The problem is to determine routes and schedules for all the trips so as to minimize the number of vehicles and travel costs while respecting network and scheduling constraints.

This route construction problem has been encountered by the authors in several contexts: e.g. school busing [7], urban transit [18], passenger trains [17], airline fleets [19]. The same problem structure also occurs in the assignment to m machines of n tasks which must be carried out within a given time interval. Bratley, Florian and Robillard [3] and Gertsbakh and Stern [11] have examined this context.

1.1 Mathematical formulation

Let P be the set of trips and I the set of intertrip arcs. An intertrip arc is considered only if it is possible to undertake trip j after trip i while respecting the time interval constraints (i.e. $a_i + t_{ij} \leq b_j$). In addition the set I is generally restricted by other a priori constraints on empty running.

This article describes the problem with a single depot where each vehicle leaves the depot only once; the nodes s and t represent exit from and entrance to the depot respectively. The network used by the vehicles is defined by the set of nodes $N = P \cup \{s,t\}$ and the set of oriented arcs $A = P \cup \{s\}\times P \cup P\times\{t\}$. Other variants with several depots, or with several exits per vehicle could also be handled.

Two types of variables are used: flow variables x_{ij}, $(i,j) \in A$ taking the value 1 when arc (i,j) is used by a vehicle, and continuous time variables t_i, $i \in P$ associated with the starting time of each trip. The variable travel costs are associated with the flow variables corresponding to the intertrip arcs, and the fixed cost per vehicle is associated with the flow variables corresponding to the arcs of the set $\{s\} \times P$.

Optimal routes respecting the scheduling constraints are the solution of the following problem:

$$\text{Min} \quad \sum_{(i,j) \in A} c_{ij} x_{ij} \tag{1}$$

$$\sum_{j \in N} x_{ij} = \sum_{j \in N} x_{ji} = 1 \qquad i \in P \tag{2}$$

$$x_{ij} \geq 0 \qquad (i,j) \in A \tag{3}$$

$$x_{ij} > 0 \Rightarrow t_i + t_{ij} \leq t_j \qquad (i,j) \in I \tag{4}$$

$$a_i \leq t_i \leq b_i \qquad i \in P \tag{5}$$

$$x_{ij} \text{ binary} \qquad (i,j) \in A . \tag{6}$$

Relations (1), (2) and (3) form a simple routing problem without scheduling constraints. This is a minimum cost flow problem having an integer solution. This network flow problem is obtained by fixing the time variables ($t_i = a_i = b_i$, $i \in P$) and was first studied by Dantzig and Fulkerson [6]. Relations (4) describe the compatibility requirements between the routes and the schedules, while constraints (5) establish the time intervals within which trips must begin.

1.2 Properties

The following proposition shows that constraints (6) can be eliminated in the formulation of the problem.

Proposition 1:

There exists an optimal integer solution to the routing problem with scheduling constraints defined by relations (1) to (5).

Proof:

The problem (1)-(5) is feasible; it is sufficient to take $(\forall i)(x_{si}=1,\ x_{it}=1,\ t_i=a_i)$. Let (X^*,T^*) be an optimal solution. If X^* is not integer, define $J = \{(i,j) | x_{ij}^* > 0\}$. Now fix T^*, and consider the least cost flow problem on the sub-network obtained by retaining only the arcs of J. This is a flow problem for which in addition to the optimal solution X^*, there exists an integer optimal solution $\bar{X}$. It is easy to check that $(\bar{X}, T^*)$ is an integer optimal solution to (1)-(5). □

Note, however, that even with proposition 1, the integer optimal solution cannot be obtained directly by linear programming as relations (4) are not linear constraints. Relations (4) can be written in the following linear form:

$$t_i + t_{ij} - t_j \leq (1 - x_{ij})\, M_{ij} \qquad (i,j) \in I \qquad (7)$$

with $M_{ij} \geq b_i + t_{ij} - a_j$. This formulation is equivalent as long as x_{ij} is required to be binary. Otherwise, for $0 < x_{ij} < 1$, we may have $0 < t_i + t_{ij} + t_j \leq (1 - x_{ij})\, M_{ij}$ which does not satisfy (4). Furthermore, it is easy to construct an example having no optimal integer solution when (4) and (6) are replaced by (7).

In the special case where $[a_i,b_i] = [0,\ |P| - 1]$ for $i \in P$, and $t_{ij} = 1$ for $(i,j) \in I$, we may set $M_{ij} = |P|$ and constraints (7) become:

$$t_i - t_j + |P|\, x_{ij} \leq |P| - 1 \qquad (i,j) \in I \qquad (8)$$

These are the subtour elimination constraints proposed by Miller, Tucker and Zemlin [14] for the travelling salesman problem. In this special case, our problem reduces to a travelling salesman problem, while in the general case, it is an m-salesman problem with time windows where the scheduling constraints ensure the elimination of subtours.

1.3 LITERATURE

The problem formed by relations (1) to (5) is non-linear and difficult to solve. Until now, large scale problems have only been solvable by approximate methods although a few ideas have been proposed for optimal treatment of this type of problem.

One possible approximate solution method is to replace each variable t_i by a set of binary variables associated with each decision to begin a trip i or not at a number of discrete points within the time interval. This method has been used by Levin [12] for air transportation, and by Swersey and Ballard [20] for school transportation; they frequently obtained an optimal integer solution using the simplex algorithm without integrality constraints. Note, however, that they restricted the objective function to include only the number of vehicles, after noting that it was more difficult to obtain integrality when travel costs were also included. Other methods proposed by Bodin et al [2] include travel costs but these require some additional approximations. Solomon [16] has also compared several heuristics for a more general problem which includes both vechicle capacity and time window constraints.

Orloff [15] sketched a rough outline of an optimal approach to the problem. He proposed the reformulation of the problem as a set partitioning problem by explicitly enumerating all routes satisfying the scheduling constraints. However, Orloff notes that the number of variables is very large even for a medium-sized problem and this approach cannot therefore be successfully applied to real problems.

Branch-and-bound methods have been most commonly used to solve fleet planning problems optimally. A relaxed problem is solved and branching is carried out to eliminate infeasible solutions. Several relaxations have been proposed; they present different compromises between solution costs and quality of approximation. For the asymmetric travelling salesman problem, Carpaneto and Toth [5] relaxed subtour elimination constraints to obtain an easily solvable assignment problem, and they improved on Bellmore and Malone's [1] branching on the flow variables to break the subtours. We adapted this method for our problem and obtained good results for small time windows [18]. Wit- larger time windows the gap between the assignment problem and the original problem is considerable and the branch-and-bound tree to be explored may become very large [9].

To reduce this gap and the size of the tree in a more general framework, Fisher and Jaikumar proposed Lagrangian relaxation while Christofides, Mingozzi and Toth [4] have proposed state-space relaxation which uses intermediate relaxations close to the original problem starting from shortest path or spanning tree problems. Magnanti's synthesis paper [13] contains the essential of these approaches.

The LP relaxation obtained by replacing (4) by its linear equivalent (6)-(7) and relaxing the integrality requirement (6) produces a smaller gap than the assignment relaxation, but the solution by the simplex method of a problem with $3*|P| + |I| = O(|P|^2)$ constraints is too costly. Our approach solves by the simplex method the LP relaxation of the set partitioning problem. This problem has only $|P|$ constraints and we use column generation to reduce the number of variables. The column generation subproblem is a shortest path with time window constraints and it does not have the integrality property when (4) is replaced by (7). If this subproblem is solved without integrality constraints, this is equivalent to the Dantzig-Wolfe decomposition of the LP relaxation with $O(|P|^2)$ constraints. We improve on this relaxation by solving the subproblem with integrality constraints using dynamic programming.

The proposed method is especially efficient when the column-generating subproblem can be quickly solved. This is the case when each vehicle performs only a few trips, or when the time intervals severely limit the set of feasible routes.

1.4 Description of specific problems examined

The algorithm was evaluated on 6 school bus transportation problems. Each problem is identified by a trip number (48,100,151) followed by AM or PM to indicate whether morning or afternoon transportation is being considered. AM and PM problems of the same size include the same trips and have time intervals of the same width, but the distribution of the intervals in time is different. Class finishing times are distributed over a longer period than starting times as shown in Table 1. The number of possible intertrip arcs is therefore greater in the afternoon, and the problems are quite different.

Table 1: Problem description

Number of trips	Number of schools	Duration of trips			Duration ($b_i - a_i$)			Duration of intertrip arcs			No.of inter-trip arcs		Total duration	
		Min.	Max.	Mean	% de 0	Max	Mean	Min.	Max.	Mean	AM	PM	AM	PM
48	6	8	33	18.7	15%	9	4.6	1	29	10.9	579	620	80	110
100	46	13	76	34.4	34%	22	7.1	1	56	19.0	1137	2088	119	166
151	55	8	76	32.9	31%	22	7.8	1	56	18.5	2620	4900	121	166

In addition to the 6 original problems, 40 variants were studied. Ten variants of the same degree of difficulty as 48 AM and 48 PM were obtained by perturbing the school schedules in a random fashion. Four more difficult variants of each of the 6 original problems were constructed by replacing the intervals $[a_i,b_i]$ by $[a_i,b_i+\Delta]$ with Δ taking values of 5, 10, 15 and 20 minutes. Each new problem of this type is identified by the original problem name followed by + Δ (e.g. 48 AM + 20). The six variants of the original problems with schedules fixed so that trip i must begin at time a_i were also studied for comparison. These are identified by adding the identification "$t_i=a_i$" to the problem name (e.g. 48 AM $t_i=a_i$).

2. SOLUTION OF THE LP RELAXATION OF THE SET PARTITIONING PROBLEM

This problem is formulated using flow variables on the routes rather than on the trips. A route satisfying the scheduling constraints (4) and (5) is specified by the trips forming it, and their starting times. We introduce the following notation:

Ω : the set of routes with schedules satisfying constraints (4) and (5);

y_r : a binary variable; 1 if route r is used, 0 otherwise;

δ_{ri}: binary constant; 1 if route r includes trip i, 0 otherwise;

c_r : cost of route r; the sum of the costs of the arcs making up the route; it includes the fixed cost per vehicle and travel costs between trips.

The solution process consists of constructing a set of least cost routes satisfying the scheduling constraints and including each trip exactly once. The set partitioning problem can be formulated as follows:

$$\text{Min} \quad \sum_{r \in \Omega} c_r \, y_r \tag{9}$$

$$\sum_{r \in \Omega} \delta_{ri} \, y_r = 1 \qquad i \in P \tag{10}$$

$$y_r \text{ binary} \qquad r \in \Omega \, . \tag{11}$$

Because of the large number of routes in Ω, the LP relaxation of the set partitioning problem is solved by column generation. A new column of minimum marginal cost is generated by solving the shortest path problem (12)-(17) with scheduling constraints defined using the dual variables σ_i of the LP relaxation of the set partitioning problem:

$$\text{Min} \quad \sum\sum_{(i,j) \in A} (c_{ij} - \sigma_i) \, x_{ij} \tag{12}$$

$$\sum_{j \in N} x_{ij} = \sum_{j \in N} x_{ji} \qquad i \in P \tag{13}$$

$$\sum_{j \in P} x_{sj} = \sum_{j \in P} x_{jt} = 1 \tag{14}$$

$$x_{ij} > 0 \Rightarrow t_i + t_{ij} \leq t_j \qquad (i,j) \in I \qquad (15)$$

$$a_i \leq t_i \leq b_i \qquad i \in P \qquad (16)$$

$$x_{ij} \text{ binary} \qquad (i,j) \in A \ . \qquad (17)$$

It can be shown using the argument of Proposition 1 that this subproblem has a binary solution even if constraint (17) is eliminated. The solution is a least cost route from s to t satisfying the time window constraints.

2.1 The Dantzig-Wolfe decomposition

The column generation solution method corresponds to a Dantzig-Wolfe decomposition where the scheduling constraints are placed in the subproblem. The network constraints (2) are divided between the subproblem and the master problem: the flow conservation constraints in the subproblem, and the requirement to undertake each trip in the master problem ($\sum_{j \in N} x_{ij} = 1, \forall i$).

According to this decomposition, the subproblem should be written without constraint (14); its solutions are thus zero or unbounded and the extreme rays are transferred to the master problem. The solution to the master problem lies in the convex cone generated by these extreme rays and does not have to satisfy the convex combination constraint. This is why this is not included in the set partitioning problem formulation (9)-(10).

On adding constraint (14), a bounded subproblem is obtained with an extreme point corresponding to each extreme ray of the unbounded subproblem. Note that if the subproblem had been bounded by the constraint $x_{ij} \leq 1$, obtained by relaxing x_{ij} binary, the solution process would not have reduced to a shortest path algorithm, but instead, a combinatorial problem as difficult as the original problem would have been obtained, i.e. the movement of several vehicles from s to t using routes with no common intertrip arcs and a total cost as negative as possible.

2.2 Solution of the subproblem

The least cost route satisfying the scheduling constraints is found using an adaptation of the Ford-Bellman algorithm which we will call the "Shortest Path with Time Windows" (SPTW)

Recall that for the classical shortest path problem, the Ford-Bellman algorithm assigns to each node a label π_i which is the cost of a path from s to i. The labels are improved by dynamic programming until optimal values are obtained; the π_i represent the cost of going from the origin s to node i by the least cost route. This algorithm compares routes according to their costs only and a single label is stored at each node because the minimal value of π_i is all that is required to calculate the minimal values of the other π_j according to the following optimality principle:

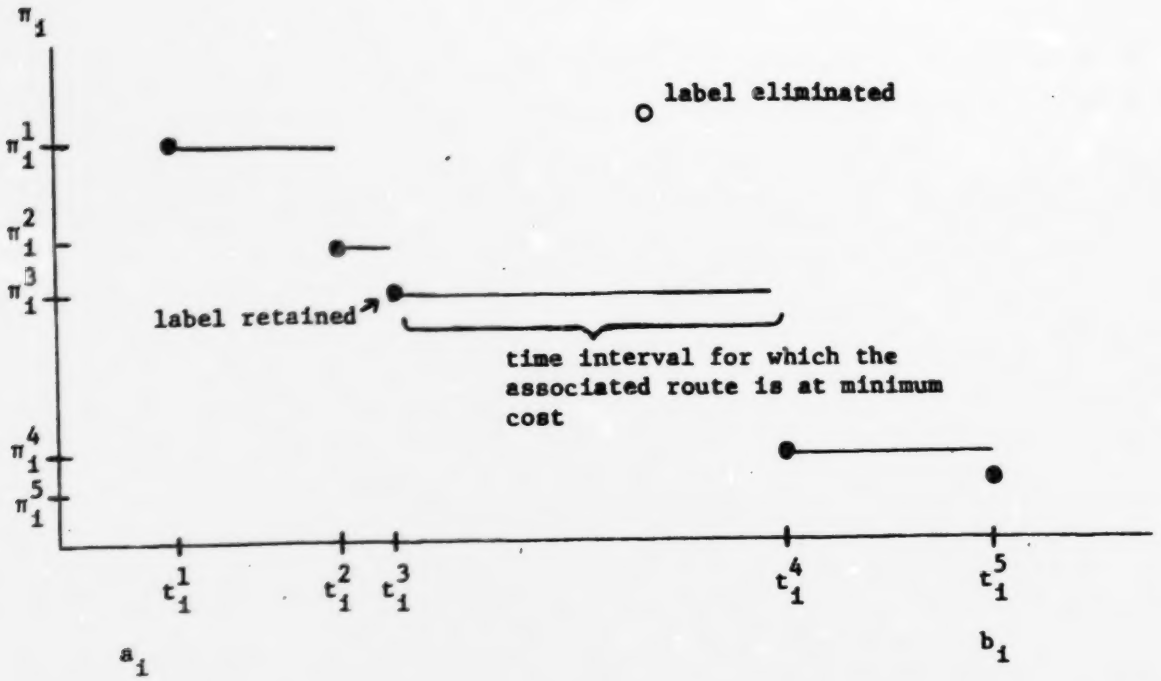
> "If r is a minimum cost route from s to j and if (i,j) is its last arc, then the subroute of r from s to i is a minimum cost route".

For the shortest path problem with scheduling constraints, routes have to be compared in terms of their arrival time at nodes, as well as costs. SPTW therefore requires a pair of labels at each node (arrival time at node i, cost of the route.) Several labels must be stored at node i as they may be required to calculate the labels at other nodes according to the following optimality principle:

> "If r is a minimum cost route from among all the routes from s to j arriving at j at time t_j or before, and if (i,j) is its terminal arc, then the subroute of r from s to i is a minimum cost route from among all routes from s to i arriving at i at time $t_j - t_{ij}$ or before".

The algorithm stores at each node the label (t_i, π_i) until a less costly route arriving at i at time t_i or before is found. The set of labels stored at node i, $\{(t_i^\alpha, \pi_i^\alpha), \alpha \geq 1\}$ contains labels for only a few times within $[a_i, b_i]$, the others having been eliminated by dominance. These labels form a list Q_i which is classified in decreasing cost order. Note as illustrated in Figure 1, that this list is also in increasing time order. List Q_j is updated sequentially from the information in Q_i by considering the routes from the origin to j composed of routes from the origin to i, and the arc (i,j). A suitable data structure and an efficient strategy for the management of the lists of lables are presented in Desrosiers, Pelletier, Soumis [8].

Figure 1: Cost vs. arrival time of labels stored at node i.



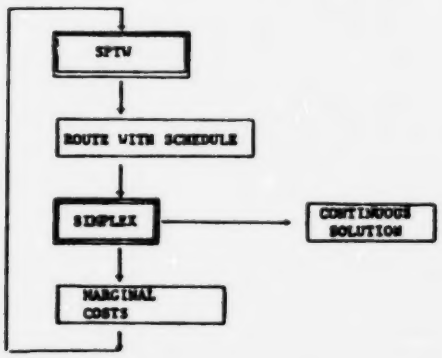
This solution method may produce routes with cycles if the network includes negative cost cycles satisfying the scheduling constraints. Such cycles are rare when the time windows are small as for a cycle to exist, it must be possible at least to make one trip, an intertrip journey and then a second trip followed by an intertrip arc bringing the vehicle back to the origin of the first trip before the end of the time interval during which this first trip must begin. If a route with a cycle is produced, it is of finite cost and duration because of the bounded time intervals. It is added to the master problem with $\delta_{ri} > 1$ for the trips used more than once. Such a route may enter the basis at a fractional level but as it cannot take the value 1, it becomes zero subsequently through an iteration of simplex or through branching.

2.3 Solution of the master problem

The simplex algorithm is used to solve the master problem. This method gives the marginal costs σ_i necessary for column generation and enables easy reoptimization each time new columns are generated. Figure 2 shows a major iteration

including the generation of routes with schedules, followed by a reoptimization of the covering problem. These iterations produce a continuous optimal solution.

Figure 2: Column generation

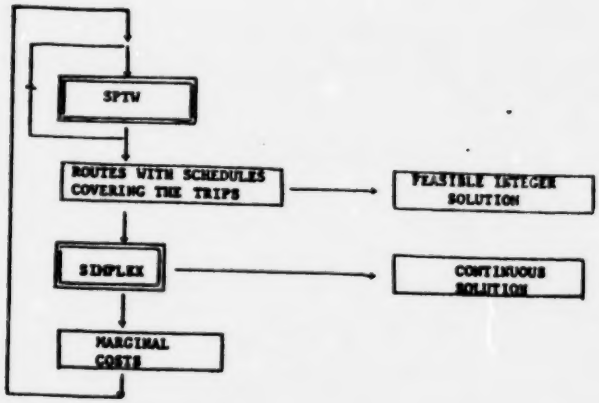


Initial trials showed that most computer time is used by the Simplex algorithm while SPTW takes very little time. The procedure was modified to accelerate convergence by generating several routes for each major interation. This modification exploits the idea that it is useful to include in the set Ω' of routes generated ($\Omega' \subseteq \Omega$), all routes with a strong probability of being optimal and even certain non-optimal routes which help reduce the objective function and improve the dual variables during the iterations. This reduces the number of coatly simplex iterations while increasing the use of SPTW, which is relatively cheap.

Several multiple column generation strategies were tried. The following method was the best of these (figure 3):

1: generate the minimum cost route satisfying the scheduling constraints;
2: remove the trips used in this route and return to 1 if any trips remain;

Figure 3: Multiple column generation at each iteration



The advantage of this method is that routes covering all trips are constructed whereas in other methods routes including only the cheapest trips are included. Furthermore, the construction of disjoint routes favours the appearance of integer solutions. Table 2 shows a comparison between simple and multiple column generation methods for the problems 48 AM, 48 PM and the 10 problems obtained by randomly modifying the school schedules.

Table 2: Comparison between two column generation strategies

	Number of major iterations	Simplex		Column generation		Total time CDC 173
		Number of iterations	Time (sec)	Number of columns	Time (sec)	
1 column/ iteration	105	761	24.5	153	4.5	29.0
Several columns /iteration	14	356	11.1	255	3.5	14.6

The second strategy considerably reduces the number of major iterations (reoptimization of the master problem.) The number of simplex iterations and the time required for these are reduced by half. Even though more columns are generated in the second method, this takes less time, as by removing the trips already used by the column generated, the size of the network is reduced for subsequent shortest path calculations.

3. OBTAINING OPTIMAL INTEGER SOLUTIONS

For this problem, the probability of spontaneously obtaining an integer solution of the master problem with the simplex method is fairly high. The generation of integer solutions is encouraged by using cuts requiring the number of vehicles and the travel costs to be rounded up to the next integer as needed. When the solution is non-integer, only a few fractional variables have to be eliminated. This is done with an implicit enumeration procedure using column generation to solve the problem at each node of the tree.

3.1 The cuts

Expression (18) separates the objective into two terms: the fixed costs associated with the number of vehicles and the travel costs:

$$C * \left(\sum_{j \in P} x_{sj} \right) + \sum_{(i,j) \in I} c_{ij} x_{ij} \qquad (18)$$

The sum of flows on the exit arcs from s towards all trips gives the number of vehicles used where C is the fixed cost per vehicle. The second term is the sum of travel costs on the intertrip arcs used (the cost of the trips is not included in the objective function.) In the applications studied, the fixed cost C is sufficiently large that it is never worthwhile to increase the number of vehicles in order to reduce travel cost. Thus, the optimal solution first minimizes the number of vehicles and for this number of vehicles, minimizes travel costs.

When the number of vehicles is non-integer in the optimal solution of the master problem, a cut is added to round this number up to at least the next integer. The master problem is then reoptimized generating new columns as needed. The efficiency of this cut is demonstrated in Table 3 where the 14 out of 40 problems in which the solution was non-integer are presented.

Table 3: Effect of the cut requiring the number of vehicles to be rounded up to at least the next integer.

Problem	Cut on vehicles	Continuous solution			First integer solution		Best integer solution		Final result		
		Buses	Cost	Time (CDC 173)	Node	Cost	Node	Cost	Nodes explored	Time (CDC 173)	Optimal
48 AM+10	without	13.3	--	38	11	232	20	229	56	502	no
	with	14	209	65	1	209	1	209	1	65	yes
48 AM+15	without	12.4	--	58	7	245	54	232	57	453	no
	with	13	205	89	6	207	6	207	14	427	yes
48 AM+20	without	11.7	--	100	9	233	12	223	26	540	no
	with	12	210	176	1	210	1	210	1	176	yes
48 PM+15	without	11.9	--	64	7	388	7	388	23	480	no
	with	12	362.8	95	5	364	10	364	14	425	yes
48 PM+20	without	11.1	--	85	10	333	16	332	22	541	no
	with	12	312.7	151	7	329	7	329	11	434	no
100 AM+ 5	without	58.5	--	38	9	408	9	408	32	348	no
	with	59	394	64	1	394	1	394	1	64	yes
100 AM+10	without	54.5	--	48	5	480	5	480	50	368	no
	with	55	475.5	79	7	480	7	480	160	426	no
100 PM	without	50.5	--	55	13	593	19	586	30	156	no
	with	51	571	118	1	571	1	571	1	118	yes
100 PM+10	without	43.5	--	118	7	647	14	639	38	437	no
	with	44	635	116	3	635	3	635	3	118	yes
100 PM+15	without	39.5	--	157	9	708	9	708	13	448	no
	with	40	688.5	206	3	692	3	692	9	404	no
151 AM+ 5	without	87.3	--	170	28	618	28	618	43	689	no
	with	88	595	183	2	595	2	595	2	185	yes
151 AM+15	without	73.5	--	251	17	810	24	808	32	978	no
	with	74	781	401	3	781	3	781	3	511	yes
151 PM+ 5	without	69.5	--	358	--	--	--	--	13	1000	no
	with	70	903	519	3	903	3	903	3	634	yes
151 PM+10	without	65.5	--	673	--	--	--	--	16	2000	no
	with	66	943	692	1	943	1	943	1	690	yes

For each problem, the first line gives the results obtained without the cut, and the second line gives the results with the cut on the number of vehicles added. The travel cost is not given when the number of vehicles is fractional, as fractions of vehicles and travel costs are not distinguished in the objective function. For the first and best integer solutions, the columns headed "node" give the number of nodes investigated before the solution was obtained. The column "nodes explored" under the final results section of the table shows the total number of nodes explored in the branching tree. All the integer solutions obtained with or without the cut are optimal for the number of vehicles; they use a number of vehicles equal to the next integer to the fractional number given by the continuous solution without the cut. The algorithm with this cut on the number of vehicles optimizes travel costs for 10 problems and for the other 4 problems the gap between the bound and last solution found is 2, 3, 4 and 16 minutes respectively. The algorithm without the cut never finds the optimal solution and the gap is over 20 minutes on average.

If the travel cost is not integer, a second cut is added requiring that this be completed to at least the next integer and the problem is reoptimized. Table 4 illustrates the effect of this second cut for the 7 problems which have fractional travel costs after using as required the cut on the number of vehicles as required.

Table 4: Effect of the cut requiring travel costs to be rounded up to at least the next integer.

Problem	Cut on travel costs	Continuous solution			First integer solution		Best integer solution		Final result		
		Buses	Cost	Time (CDC 173)	Node	Cost	Node	Cost	Nodes explored	Time (CDC 173)	Optimal
48 PM+5	without	14	380.6	33	6	389	30	387	64	450	?
	with	14	381	33	12	388	15	387	50	711	?
48 PM+10	without	13	370.2	52	6	373	6	373	44	450	?
	with	13	371	61	3	373	3	373	30	576	?
48 PM+15	without	12	362.8	93	5	366	10	364	14	423	yes
	with	12	363	101	3	364	3	364	10	334	yes
48 PM+20	without	12	312.7	151	7	329	7	329	11	434	no
	with	12	313	161	9	318	9	318	16	755	?
100 AM+10	without	55	475.3	79	7	480	7	480	160	426	no
	with	55	476	83	2	480	11	479	153	827	?
100 PM+15	without	40	688.5	173	3	692	3	692	9	404	no
	with	40	689	177	3	690	3	690	76	897	?
151 PM	without	75	850.3	283	5	860	18	854	24	1189	no
	with	75	851	301	2	851	2	851	2	378	yes

For each problem, the first line presents the results obtained using as needed the cut on the number of vehicles, while the second line presents the results when the cut on travel costs is added. Using the second cut, the first integer solution is usually obtained with less exploration, and the solution obtained is better. The comparison between the best integer solutions also favours the method with two cuts. However, even with both cuts, several problems could not be solved optimally although the solutions obtained were close to optimal.

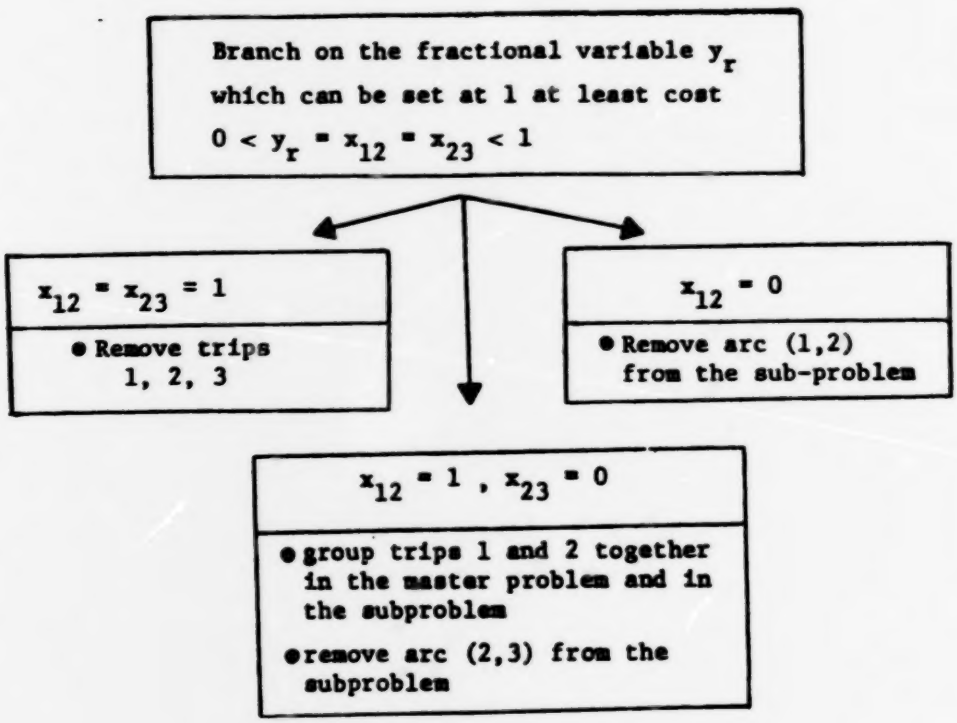
The two cuts can be placed in the simplex tableau from the beginning as the coefficients of the variables are known. The right-hand-sides are first fixed at zero so that the cuts are inactive. When one of the cuts is required the right-hand-side is increased to the desired value; in the same way, it is decreased when the cut is dropped. This is useful because the second cut has to be tightened when descending into the tree fixing variables, and then loosened on moving back up to explore other nodes.

3.2 The Branching Tree

The branching at a node takes place on the route with fractional flow for which this flow can be set at 1 at least additional cost, according to the estimates given by the marginal costs. Bellmore and Malone's [1] method as adapted by Carpaneto and Toth [5] for the travelling salesman problem is used for the branching. This is carried out not by fixing the y_r variables, but by setting at 0 or 1 the k variables x_{ij} associated with the arcs which make up the route r. The 2^k ways of fixing these variables are grouped into k + 1 branches including respectively 1, 1, 2, 4,..., 2^{k-1} ways. The first branch fixes the k arcs at 1. The second fixes the first k-1 arcs at 1 and the last at 0. The third fixes the first k-2 arcs at 1 and the next one at 0, and so on until the k + 1st branch where the first arc is set at 0. This branching procedure retains in each branch a master problem and subproblem of the same type as those initially defined. Branching by setting $y_r = 0$ would have destroyed the shortest path structure of the subproblems.

The branching proposed does not apply to routes with a single trip as the number of arcs k is zero. These routes may appear in the problem as slack variables and they may be fractional. This difficulty is circumvented by branching only on fractional variables corresponding to routes with two or more trips. When these variables are all integer, the slack variables will also be integer. To simplify the explanation of the branching method and the processing at each node, an example is given in which branching takes place on a route using trips 1, 2 and 3 in that order. Figure 4 shows the branches considered.

Figure 4: Implicit enumeration tree


Branch on the fractional variable y_r which can be set at 1 at least cost
$0 < y_r = x_{12} = x_{23} < 1$
$x_{12} = x_{23} = 1$
• Remove trips 1, 2, 3
$x_{12} = 0$
• Remove arc (1,2) from the sub-problem
$x_{12} = 1$, $x_{23} = 0$
• group trips 1 and 2 together in the master problem and in the subproblem
• remove arc (2,3) from the subproblem

In the implicit enumeration tree, the leftmost node is always explored first. This reduces the size of the problem and produces an integer solution more quickly.

3.3 Processing at each branch

For the left branch $x_{12} = x_{23} = 1$, the solution includes a route using the trips 1, 2 and 3. To find the other routes, rows 1, 2 and 3 are removed from the master problem, as are all the routes containing one of these trips. All the nodes corresponding to these trips are removed from the subproblem. The covering problem is reoptimized generating new routes as needed.

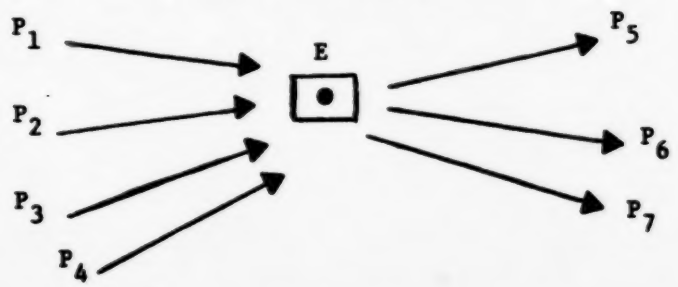
For the central branch $x_{12} = 1$, $x_{23} = 0$, the solution includes a route using the trips 1 and 2, but not the trip 3. This route can be completed by adding other trips. To complete this route, and to find the other routes forming a solution, rows 1 and 2 of the master problem are combined and called row 1∧2. Of all the routes containing either trip 1 or trip 2, only those containing both trip 1 and trip 2 and excluding trip 3 are retained. These routes are assigned the value 1 on row 1∧2 of the master problem. Similarly, in the subproblem, trips 1 and 2 are combined. The new trip 1∧2 has the associated time interval $[a_1, \min\{b_1, b_2 - t_{12}\}]$. Arcs which arrived at 1 now arrive at 1∧2 and those arriving at 2 are eliminated. Arcs leaving 1 are eliminated as is the arc (2,3). Other arcs leaving 2 now leave 1∧2 with costs and durations increased by c_{12} and t_{12} respectively.

For the right branch $x_{12} = 0$, the solution contains no routes with trip 1 followed by trip 2. All such routes are eliminated from the master problem and the arc (1,2) is removed from the subproblem to avoid generation of others.

4. NETWORK REDUCTION

There are of the order $|P|^2$ intertrip arcs in the original network. This number is reduced by using school nodes as illustrated in figure 5. Note that this reduction does not affect the optimality of the solution.

Figure 5: School network aggregation



The example shows an afternoon problem where P_5, P_6 and P_7 are all trips leaving school E, and where P_1 to P_4 are the trips which could finish soon enough to be followed by one of the trips beginning at school E. Intertrip arcs between $\{P_1, P_2, P_3, P_4\}$ and $\{P_5, P_6, P_7\}$ are replaced by arcs from $\{P_1, P_2, P_3, P_4\}$ to E and arcs from E to $\{P_5, P_6, P_7\}$. This reduction produces a maximum number of arcs for this school of $3 + 4 = 7$, instead of the $3 \times 4 = 12$ arcs in the original network. The savings obtained by replacing a product by a sum are obviously greater when the numbers are large.

Note that if intertrip arcs (1,5) and (1,6) exist while (1,7) is infeasible ($a_1 + t_{17} > b_7$) in the original network, the reduction procedure adds a path from 1 to 7. However, the shortest path calculation with scheduling constraints is not affected, because this new path arrives at trip 7 after b_7 and is not considered. Note also that during the implicit enumeration it is not possible to fix an intertrip arc at 0 in the reduced network. To set arc (1,5) at 0, arcs (p_1, E) and (E, p_5) must be removed and the intertrip arcs from $\{p_1\}$ to $\{p_6$ and $p_7\}$ and from $\{P_2, P_3, P_4\}$ to $\{P_5\}$ must be re-established. The savings in time and memory in SPTW due to reduction dominate by a wide margin the additional costs of adding arcs during implicit enumeration, given the small number of nodes explored. Table 5 shows the number of arcs before and after aggregation for the problems studied.

Table 5: Effect of aggregation on the size of the network

Problem name	Original problem		Variants of the problem							
			$\Delta = 5$		$\Delta = 10$		$\Delta = 15$		$\Delta = 20$	
48 AM	579[1]	133[2]	669	145	764	160	876	177	999	193
48 PM	620	135	711	143	800	157	888	169	999	180
100 AM	1137	657	1532	814	1793	927	2093	1060	2505	1244
100 PM	2088	971	2478	1112	2797	1232	3018	1322	3338	1462
151 AM	2620	1114	3572	1392	4311	1635	5108	1899	6068	2240
151 PM	4900	1706	5787	1938	6529	2151	7077	2322	7795	2568

(1) Number of arcs before aggregation

(2) Number of arcs after aggregation

5. TESTS

The results are divided into two parts: the results for the six original problems and then the results of study of the effect of widening the time windows $[a_i, b_i]$.

5.1 The original problems

These problems were easily solved because the time intervals are fairly narrow. Larger problems could have been solved if the memory limit of 60K words (60 bits/word) had not been reached. To handle larger problems, the Land-Powell simplex algorithm used could be replaced by an algorithm which stores the inverse basis in product form rather than explicitly.

Table 6: Results for the 6 original problems

Problem	Columns generated	Labels per trip	Nodes explored	Calculation time	Optimal solution		Savings in vehicles
					Buses	Cost	
48 AM	163	1.5	6	16/4.1	17	191	15%
48 PM	156	2.	1	13/3.2	17	320	29%
100 AM	238	1.6	1	12/14	63	390	10%
100 PM	290	2.1	1	35/24	51	571	14%
151 AM	393	2.	1	45/49	94	560	13%
151 PM	473	2.	2	307/69	75	851	18%

Note that all these problems have an integer solution with the same number of buses and the same cost as for the bound determined by relaxing integrality with the cuts. The implicit enumeration tree contains few nodes and its size does not increase with the number of trips. The number of columns generated in the master problem increases linearly with the number of trips. The average number of labels generated for each trip in the network by the shortest path algorithm with scheduling constraints is very small and does not increase with the size of the network. The computing time in seconds on the CDC 173 (written as time in simplex/time in shortest path) does not increase exponentially with problem size. The exceptional value of 307 seconds for 151 PM is due to the large number of reinversions carried out. The last column gives the savings in the number of vehicles

needed in relation to the fleet size required to make the trips with a fixed schedule. These savings are remarkably high even though the time intervals $[a_i, b_i]$ are relatively narrow.

5.2 Effect of time interval width $[a_i, b_i]$

Table 7 gives the results for variants of the problems in which the intervals $[a_i, b_i]$ were gradually widened until they reached a width of roughly one trip duration when $\Delta = 20$. These variants are much more difficult, the number of arcs is almost doubled for $\Delta = 20$ and the difference between these problems and the basic network problem where $t_i = a_i$ increases, as indicated by the reductions of up to 50% in the number of vehicles used. Computing time increases with interval width but most problems are still solved optimally. For those where optimality is not proven, the number of buses is minimized, and the gap on the travel cost is less than 1.6%. Note that the problems where optimality is not proven are not the largest problems nor those with the widest intervals $[a_i, b_i]$. The performance of the algorithm varies with the width of the intervals as follows. The number of nodes explored in the enumeration tree before reaching the best integer solution hardly increases at all. The number of columns generated increases roughly in proportion to the number of arcs. The number of labels per trip in the shortest path algorithm with scheduling constraints increases, but stays relatively small. The computing time in the simplex increases faster than the computing time in the shortest path.

Note, however, that if the time intervals $[a_i,b_i]$ are further increased, solution time increases significantly. In fact, many routes with cycles which satisfy the scheduling constraints are generated, and the number of labels per node increases greatly. Thus, this approach is not suitable for solving efficiently an m-TSP where all nodes must be visited within the time interval $[a_i,b_i] = [0,|P|-1]$.

Table 7: Effect of time interval width $[a_i, b_i]$

Problem	Number of arcs	Simplex + cuts			Best integer solution						Final results		
		Buses	Cost	Computing time[1]	Buses	Cost	Node	Optimal	Sav.(3) in vehicles	Nodes expl. (4)	Columns[5]	Labels[6]	Computing time
48 AM $t_i=a_i$	490	20	185	3.1/1.6	20	185	1	yes	--	1	129	1	3.1/1.6
48 AM	579	17	191	5.3/3.1	17	191	6	yes	15%	6	163	1.5	16.5/4.1
48 AM+5	669	15	213	13/5.3	15	213	2	yes	25%	2	174	2	19.1/5.7
48 AM+10	764	14	209	23/15	14	209	2	yes	30%	2	262	5	64/18
48 AM+15	876	13	205	33/25	13	207	6	?	35%	26	424	9	373/43
48 AM+20	999	12	210	44/56	12	210	1	yes	40%	1	307	14	84/71
48 PM $t_i=a_i$	512	24	212	2.6/1.3	24	212	1	yes	--	1	121	1	2.6/1.3
48 PM	620	17	320	13/3.2	17	320	1	yes	29%	1	156	2	13/3.2
48 PM+5	711	14	381	25/7.8	14	387	15	?	42%	30	431	3.4	687/24
48 PM+10	800	13	371	36/16	13	373	3	?	46%	30	402	5.4	526/40
48 PM+15	888	12	363	35/29	12	364	3	yes	50%	10	365	7.8	274/60
48 PM+20	999	12	313	34/51	12	318	9	?	50%	16	404	11	668/87
100 AM $t_i=a_i$	787	70	286	5.5/4.6	70	286	1	yes	--	1	181	1	5.5/4.6
100 AM	1137	63	390	12/14	63	390	1	yes	10%	1	238	1.6	12/14
100 AM+5	1532	59	394	18/20	59	394	1	yes	11%	1	260	1.8	31/21
100 AM+10	1793	55	476	24/24	55	479	11	?	21%	153	420	2	756/71
100 AM+15	2093	51	515	53/36	51	515	2	yes	27%	2	299	2.5	67/36
100 AM+20	2505	46	608	59/57	46	608	3	yes	34%	3	337	3.5	80/63
100 PM $t_i=a_i$	1703	59	464	4.9/5.8	59	464	1	yes	--	1	175	?	4.9/5.8
100 PM	2088	51	571	33/22	51	571	1	yes	14%	1	290	2.1	35/24
100 PM+5	2478	46	644	41/32	46	644	1	yes	22%	1	308	2.1	41/32
100 PM+10	2797	44	635	74/44	44	635	2	yes	25%	2	347	2.6	185/47
100 PM+15	3018	40	689	115/55	40	690	3	?	37%	76	473	3	807/90
100 PM+20	3338	37	736	176/76	37	736	77	yes	37%	7	455	3.1	379/102
151 AM $t_i=a_i$	1535	108	406	11/10	108	406	1	yes	--	1	261	1	11/10
151 AM	2620	94	560	45/49	94	560	1	yes	13%	1	393	2	45/49
151 AM+5	3572	88	595	90/80	88	595	1	yes	19%	1	427	2.3	128/82
151 AM+10	4311	82	734	263/119	82	734	1	yes	24%	1	454	3	263/119
151 AM+15	5108	74	781	183/68	74	781	3	yes	31%	3	518	3.2	319/192
151 AM+20	6068		(memory > 200 000$_8$)										
151 PM $t_i=a_i$	3769	92	654	14/18	92	654	1	yes	--	1	298	1	14/18
151 PM	4900	75	850.3	215/67	75	851	2	yes	18%	2	473	2	307/69
151 PM+5	5787	70	903	354/104	70	903	6	yes	24%	6	525	2.3	648/122
151 PM+10	6529	66	943	637/133	66	943	1	yes	28%	1	546	3.5	557/135
151 PM+15	7077	58	1059	633/261	58	1059	4	yes	37%	4	637	4.2	877/269
151 PM+20	7795		(memory > 200 000$_8$)										

1- COMPUTING TIME ON CDC 173, SECONDS FOR SIMPLEX/SECONDS FOR SPTW
2- NUMBER OF NODES ENUMERATED BEFORE REACHING BEST SOLUTION
3- SAVINGS IN NUMBER OF VEHICLES COMPARED WITH FIXED SCHEDULE TRIPS
4- TOTAL NUMBER OF NODES ENUMERATED
5- TOTAL NUMBER OF COLUMNS GENERATED
6- AVERAGE NUMBER OF LABELS/TRIP IN SPTW

6. CONCLUSION

By handling the scheduling constraints during the construction of routes with the SPTW algorithm, and by solving the covering problem with simplex, the integrality of the solution is lost, but an excellent lower bound is obtained. This bound does not deteriorate as the number of trips increases nor as the time intervals are widened. It gives the optimal number of vehicles for all problems examined and the minimum trip time 4 times out of 5; in other cases the gap is of the order of 1%. It is easy to find an integer solution subsequently. The simplex method often produces an integer solution directly, or this is obtained by reoptimizing with the simplex method with several variables fixed. In the few cases where the first integer solution had a cost higher than the simplex bound, the Bellmore and Malone branching method was inefficient in improving the solution or proving optimality. The tree to be enumerated was very large even though the gap was very small.

The algorithm was not used for problems with more than 151 trips because of the 60K words available computer memory limit. Larger problems could be solved as the computing time does not increase exponentially with the number of trips. This would be possible using a different computer or a simplex algorithm which stores the inverse basis in product form.

For problems with relatively wide time windows, a reduction of up to 50% in fleet size is obtained as compared with the fleet size required to make the same trips but with fixed schedules. Methods which first solve the network problem with fixed schedules and then modify the network using branch-and-bound to satisfy the scheduling constraints were found to be inefficient for this type of problem. Although the work required at each node of the branch-and-bound tree is less, and the solutions obtained are integer, only small problems could be solved. The initial solutions were a long way from optimality and the size of the tree required to improve these increased rapidly with the size of the problem, and with the width of the time intervals.

The algorithm developed opens up the possibility of reducing the number of vehicles required to carry out a set of trips. Large problems with fairly wide time windlows can be solved allowing substantial savings to be made.

REFERENCES

[1] Bellmore, Malone, "Pathology of Travelling Salesman Subtour Elimination Algorithm" Operations Research 19, 1971, 278-307.

[2] Bodin, Golden, Assad, Ball, "Routing and Scheduling of Vehicules and Crews: The State of the Art", Computer and Operations Research 10, 1983, 69-211.

[3] Bratley, Florian, Robillard, "Scheduling with Earliest Start and Due Date Constraints on Multiple Machines" Naval Res. Logistics Quat., vol. 22, 1975, 165-173.

[4] Christofides, Mingozzi, Toth, "State-Space Relaxation Procedures for the Computation of Bound to Routing Problems" Networks, vol. 11, 1981, 145-164.

[5] Carpaneto, Toth, "Some New Branching and Bounding Criteria for the Asymmetric Traveling Salesman Problem", Management Science, 26(7), 1980, 736-743.

[6] Dantzig, Fulkerson, "Minimizing the Number of Tankers to Meet a Fixed Schedule", Naval Res. Logistics Quat., vol. 1, 1954, 217-222.

[7] Desrosiers, Ferland, Rousseau, Lapalme et Chapleau, "TRANSCOL: A Multi-Period School Bus Routing and Scheduling System", Publication #164, Centre de recherche sur les transports, Montréal, 1980.

[8] Desrosiers, Pelletier, Soumis, "Plus court chemin avec contraintes d'horaires", RAIRO, 17(4), 1983, 1-21.

[9] Desrosiers, Soumis, Desrochers, Sauvé, "Routing and Scheduling with Time Windows Solved by Network Relaxation and Branch-and-Bound on Time Variables", Computer Scheduling of Public Transport, vol. II, North Holland, J.-M. Rousseau ed., to appear.

[10] Fisher, Jaikumar, "A Generalized Assignment Heuristic for Vehicle Routing", Networks 11, 1981, 109-124.

[11] Gertsbakh, Stern, "Minimal Resources for Fixed and Variable Job Schedules" Operations Research, vol. 26, 1978, 68-85.

[12] Levin, "Scheduling and Fleet Routing Models for Transportation Systems" Transportation Science Vol. 5, 1971, 232-255.

[13] Magnanti, "Combinatorial Optimization and Vehicle Fleet Planning: Perspectives and Prospects", Networks 11, 1981, 179-213.

[14] Miller, Tucker, Zemlin "Integer Programming Formulation of Travelling Salesman Problems" Journal of the Association of Computing Machinery 7, 1960, 326-332.

[15] Orloff, "Route Constrained Fleet Scheduling" Transportation Science, Vol. 10, No. 2, 1976, 149-168.

[16] Solomon, "Vehicle Routing and Scheduling with Time Window Constraints: Models and Algorithms", Working Paper #83-02-01, the Warton School, University of Pennsylvania.

[17] Soumis, Desrosiers, "Locomotive Maintenance Scheduling and Train Assignment" Publication #239, Centre de recherche sur les Transports, Montréal, 1983.

[18] Soumis, Desrosiers, Desrochers, "Optimal Urban Bus Routing with Scheduling Flexibilities", Proceedings of the 11th IFIP Conference on System Modelling and Optimization, P. Thoft-Christensen ed., to appear.

[19] Soumis, Ferland, Rousseau, "A Model for Large Scale Aircraft Routing Problems", Transportation Research B, Vol. 14, No. 1, 1980, 181-201.

[20] Swersey, Ballard, "Scheduling School Buses" Published at Yale School of Organization and Management and presented at TIMS/ORSA Conference San-Diego 1982.

POOR

QUALITY

ORIGINAL